Homework 3: Support Vector Machines and Boosting

Devin Quirozoliver
arik182@cs.pdx.edu, 904032321

June 5, 2013

1 Introduction

The following is a brief analysis of an SVM classifier before and after boosting using the adaboost algorithm. The SVM classifier is svm_light, which is available at `http://svmlight.joachims.org` for download. The data set provided to the classifier represents a collection of email containing a certain quantity of spam. The dataset can be found at `http://web.cecs.pdx.edu/~mm/MachineLearningSpring2013/spam.zip` for download. Three experiments were performed. During the first, svm_light was simply used to classify the data, and this was analyzed using the ROC.py script. During the second experiment, the adaboost algorithm was applied to results from the original classifier in order to boost the accuracy of the experiment. 10 iterations of boosting were used to increase performance of the classifier. During the third experiment the adaboost algorithm was also used, this time with 20 boosting iterations.

2 Setup

The source code for this program was written in python, and can be found on my github account at `https://github.com/arik181/adaboost\_experiment.` In order to run this experiment, you will need python 2.7. Compatibility with other versions of python has not been tested.

3 Overview

4 Experiment 1

From the ROC curve, we can tell that the classifier clearly performed more strongly on negative cases than on positive cases in the test data. The classifier performed reasonably well, as expected, with 90% accuracy on the test data, 82% recall and 82% precision.

Experiment 1 statistics	
TP	1198
FP	256
TN	2101
FN	126
P	1454
N	2227
FPR	0.11
TPR	0.90
Precision	0.82
Recall	0.82
Accuracy	0.90

5 Experiment 2

During this second experiment, the adaboost algorithm was used to boost the performance of the svm_light algorithm. 10 boosting iterations were performed in total, and 10 weak hypothesis were generated in order to create an ensemble hypothesis. Unfortunately, the new hypothesis performed worse, on average than the svm_light classifier working alone. In fact, I am uncertain whether the performance of adaboost in this instance was significantly better than a selection at random for each feature from the set of hypothesis used by adaboost.

There are a number of possible explanations for this behavior. First, the algorithm could be represented incorrectly in the code. During the construction of the adaboost implementation, two different normalization factors were used, first a simple average across the data, and then the normalizer Z_t, represented below. Neither of these seemed to improve the performance of the classifier. Another possible cause for the weakness of the adaboost implementation is the use of a strong algorithm. It has been noted that the adaboost algorithm performs better when given a collection of comparatively weak classifiers. The classifiers used in the boosting algorithm were very similar in terms of their accuracy and precision, deviating no more than plus or minus 5%. A third possible cause of the weakness, perhaps the most likely, is a poor implementation of the change in weights prior to boosting.

Correct classifications	
H correct	3207
h[0] correct	3245
h[1] correct	3182
h[2] correct	3257
h[3] correct	3271
h[4] correct	3169
h[5] correct	3296
h[6] correct	3219
h[7] correct	3241
h[8] correct	3298
h[9] correct	3205

Experiment 2 statistics	
TP	1313
FP	333
TN	1894
FN	141
P	1646
N	2035
FPR	0.16
TPR	0.80
Precision	0.80
Recall	0.80
Accuracy	0.87

Figure 1: h[t] represents each of ten weak classifiers used in training the ensemble hypothesis H

6 Experiment 3

The third experiment saw an increase from 10 iterations to 20 iterations of the boosting algorithm. The algorithm performance was similar to that of the second experiment, with little difference between the accuracy of the classifiers.

Correct classifications	
H correct	3212
h[0] correct	3264
h[1] correct	3225
h[2] correct	3292
h[3] correct	3267
h[4] correct	3259
h[5] correct	3234
h[6] correct	3251
h[7] correct	3261
h[8] correct	3213
h[9] correct	3258

Experiment 3 statistics	
TP	1316
FP	331
TN	1896
FN	138
P	1647
N	2034
FPR	0.16
TPR	0.80
Precision	0.80
Recall	0.80
Accuracy	0.87

Figure 2: h[t] represents each of ten weak classifiers used in training the ensemble hypothesis H

7 Summary and Observations

This experiment concerned the use of the adaboost boosting algorithm to increase the performance of an SVM classifier in classifying spam. The most notable characteristic of this experiment was its failure to reproduce the performance boost associated with the algorithm. This could be attributed to an error in implementing the algorithm, or it could be a quality of the boosting algorithm when used with strong classifiers.

My personal feeling is that this approach should not make the results of the ensemble classifier worse than the component weak classifiers, no matter the type or accuracy of classifier used, and this leads me to believe that there is a problem in my implementation, rather than an inherent deficiency in the boosting algorithm itself.